Data Mining Lab, Exercise 5

Team# 1: Poliakov Valerii, Holovnia Dmytro, Selvaraj Sinju

Datasets: 1_heart_disease.csv, new records.csv

Tasks:

1. Make research how the size of a decision tree influences the classification error rate. To create a decision tree, apply either CART or C4.5 algorithm. Using the k-fold cross validation with k chosen between 2 and 10, determine the error rate on the test set and on the training set. The details of the computations should be presented in the table given below. Discuss the results. Comment on the overfitting if occurs.

2. Classify new records, provided in file “new records.txt”, using the decision trees created by the CART and C4.5 algorithms trained on the whole data set.

Task# 1 Make research how the size of a decision tree influences the classification error rate

Step 1. Read the 1_heart_disease.csv to understand the data.

```
> library("rpart");library("rpart.plot");library("C50"); library(class)
> d<-read.csv(file="1_heart_disease.csv", stringsAsFactors=TRUE)
```

Step 2. Create folds for k-fold cross validation.

```
> nbreaks<-10
> folds <- cut(seq(1,nrow(d)), breaks=nbreaks, labels=FALSE)
```

Step 3. Data is ordered so we should randomly shuffle the data.

```
> d<-d[sample(nrow(d)),]
```

Step 4. Running cycle from 1 to 20 with step 2. This is needed to test tree size.

```
> for(i in seq(1, 20, by=2)){
```

Step 5. Another cycle inside the first one needed for k-fold cross validation.

```
> for(k in seq(1:nbreaks)){
```

Step 6. Set test and train data using folds.

```
> test_indices <- which(folds==k,arr.ind=TRUE)
> d.train <- d[-test_indices, 1:13]
> d.train.class <- d[-test_indices, 14]
> d.test <- d[test_indices,1:13]
> d.test.class<- d[test_indices, 14]
```

Step 7. Get trained model using fold data.

```
> model<-C5.0(x=d.train, y=d.train.class, control=C5.0Control(CF=1.0,
minCases = i))
```

Step 8. Predict train and test data and get errors for both.

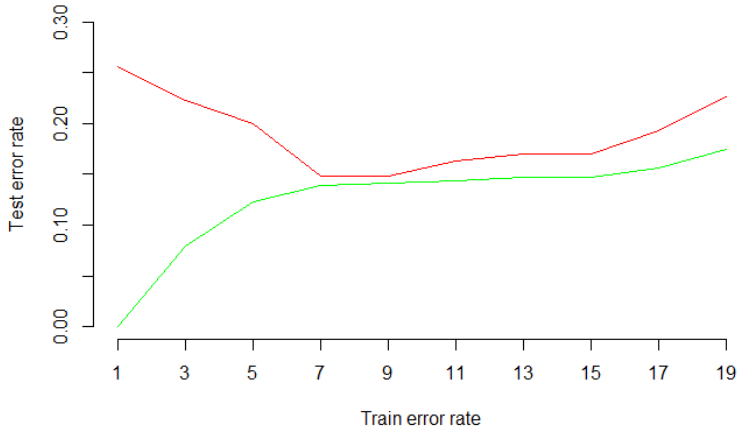
```
> pred <- predict(model, d.test, type="class")
> test_av_err_rate = test_av_err_rate + mean(pred != d.test.class)
>
> pred <- predict(model, d.train, type="class")
> train_av_err_rate = train_av_err_rate + mean(pred != d.train.class)
```

Step 9. Show the results to fill in the table.

```
> cat("minCases: ", i, " | Tree size: ", model$size)
> cat(" | test_error_rate: ", test_av_err_rate/nbreaks)
> cat(" | train_error_rate: ", train_av_err_rate/nbreaks, "\r\n")
```

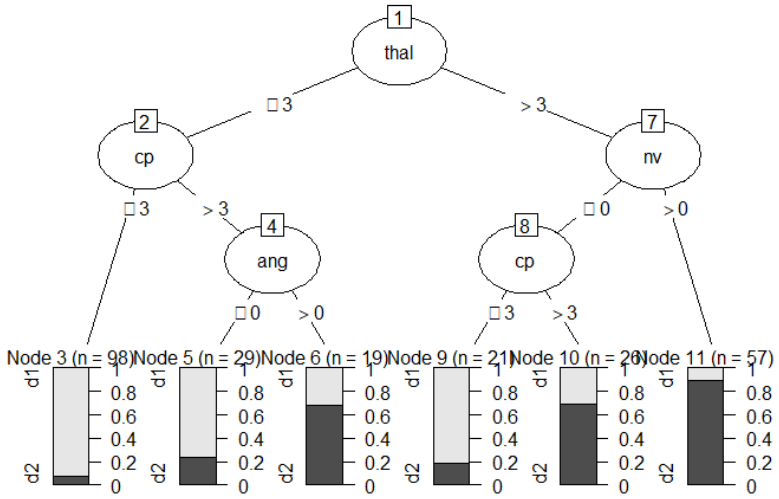
Step 10. Results plot.

```
> plot(ti, taer, type="l", ylim=c(0, 0.3), ylab="Test error rate", xlab="Train error
rate", axes = FALSE, col="red")
> lines(ti, traer, col="green")
> xlabel <- seq(1, 20, by = 2)
> axis(1, at = xlabel)
> axis(2)
```



For small values of minCases, we observe the overfitting: the model gives small average error for the training data and big average error for the test data, which means that it is “fitted” to the training data but does not work well for new data. The best average error rate is achieved for smaller models. In our case best value is spotted with minCases = 7.

C4.5 algorithm result tree with minCases = 19



The experiment details:
Chosen algorithm: C4.5
Number of folds, k, in the k-fold cross validation: 10

Table. The results of the experiment

Pruning parameter value *(minsplit or minCases)*	Decision tree size	Error rate on the test sets	Error rate on the training sets
1	41	0.2555556	0
3	17	0.2222222	0.07901235
5	14	0.2	0.1230453
7	6	0.1481481	0.1395062
9	6	0.1481481	0.1415638
11	6	0.162963	0.1436214
13	6	0.1703704	0.1465021
15	6	0.1703704	0.1465021
17	6	0.1925926	0.1559671
19	4	0.2259259	0.1744856

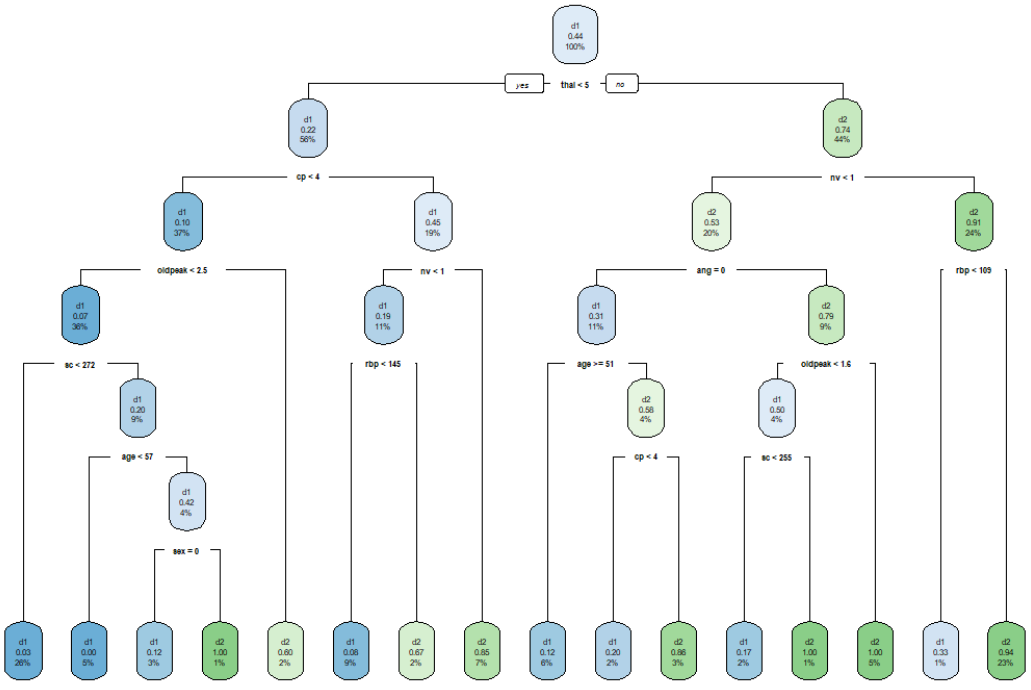
Task# 2 Classify new records

Step 1. First, we should read new records.

```
> n<-read.csv(file="new records.csv", header=TRUE)
> n
```

Step 2. Get trained model using CART algorithm and output a plot with tree.

```
> model<-rpart(class ~ ., data = d, method = "class", control=rpart.control(minsplit
=7, cp=0.0))
> rpart.plot(model)
```



Step 3. Predict class for new data.

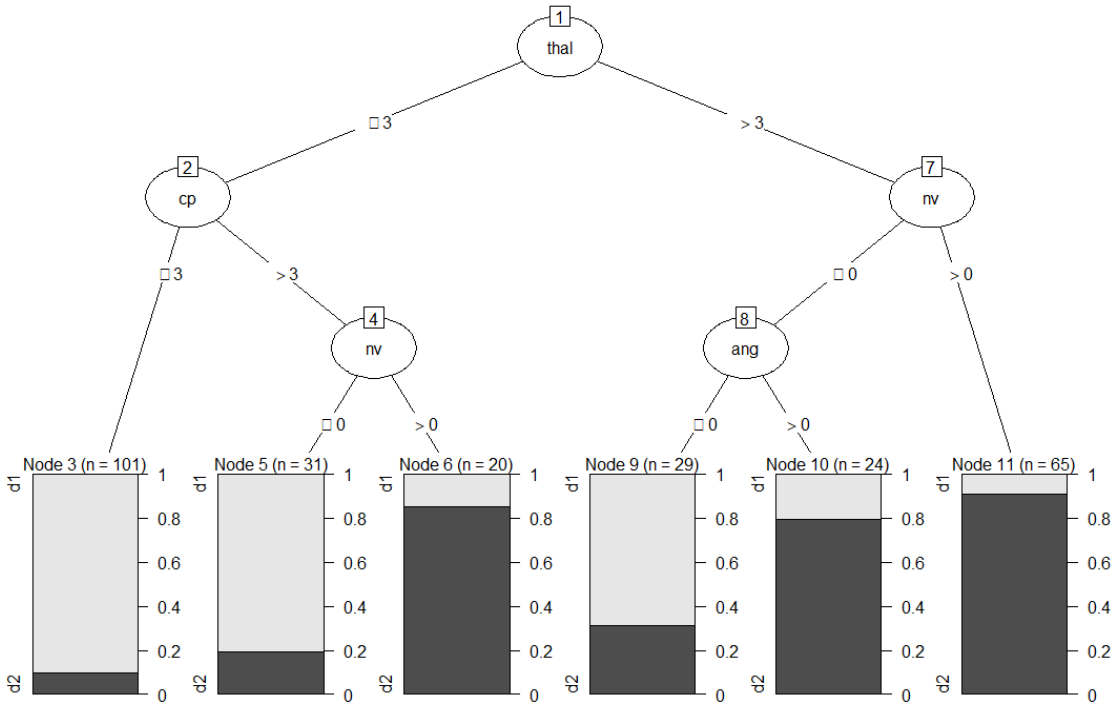
```
> pred <- predict(model, n, type="class")
> pred
```

1	2	3
d1	d2	d2

Step 4. Get trained model using C4.5 algorithm and output a plot with tree.

```
> d.train <- d[, 1:13]
> d.train.class <- d[, 14]
```

```
> model<-C5.0(x=d.train, y=d.train.class, control=C5.0Control(CF=1.0, minCases = 7))
> plot(model)
```



Step 5. Predict class for new data.

```
> pred <- predict(model, n, type="class")
> pred
```

1	2	3
d1	d2	d2

As we can see from the result tables both algorithms have the same result.